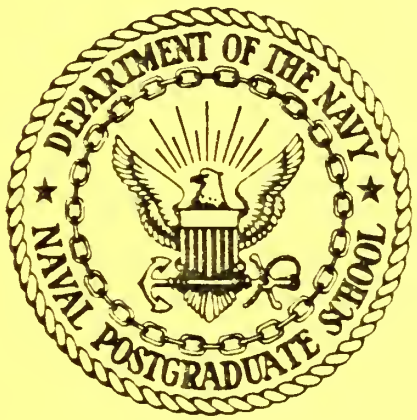
NPS52-84-010

NAVAL POSTGRADUATE SCHOOL
Monterey, California



INFORMATION-THEORETIC PROPERTIES
OF LANGUAGES AND THEIR GRAMMARS

B. J. MacLennan

August 1984

Approved for public release, distribution unlimited

Prepared for:

Chief of Naval Research
Arlington, VA 22217

FedDocs
D 208.14/2
NPS-52-84-010

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Commodore R. H. Shumaker
Superintendent

D. A. Schrady
Provost

The work reported herein was supported by the Office of Naval Research under contract number N00014-84-WR-24087.

Reproduction of all or part of this report is authorized.

This report was prepared by:

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NPS52-84-010	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Information - Theoretic Properties of Languages and their Grammars		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) B. J. MacLennan		8. CONTRACT OR GRANT NUMBER(s) N00014-84-WR-24087
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research Arlington, Virginia 22217		12. REPORT DATE August 1984 13. NUMBER OF PAGES 40
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

entropy, negentropy, information theory, language metrics, quantitative linguistics, language complexity, information content of languages, average derivation length, average information used by grammars, average string length, information density, frequency of occurrence, annotated grammar.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

We describe means for computing a number of information-theoretic properties of languages and their grammars. For example, the entropy of a system of symbols is widely recognized as a measure of that system's complexity and organization. We show how the entropy of a language can be computed in a simple way from a grammar annotated with production probabilities. We then develop means for statistically estimating these production probabilities from measurable properties of strings in the language. We also consider the computation of other information theoretic properties of languages and grammars, such as the average infor-

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DUDLEY KNOX LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY CA 93943-5101

mation born by a symbol in a language and the average information used by the productions of a grammar.

Information-Theoretic Properties of Languages and their Grammars

Bruce J. MacLennan

Computer Science Department

Naval Postgraduate School

Monterey, CA 93943

Abstract: We describe means for computing a number of information-theoretic properties of languages and their grammars. For example, the *entropy* of a system of symbols is widely recognized as a measure of that system's complexity and organization. We show how the entropy of a language can be computed in a simple way from a grammar annotated with production probabilities. We then develop means for statistically estimating these production probabilities from measurable properties of strings in the language. We also consider the computation of other information theoretic properties of languages and grammars, such as the average information born by a symbol in a language and the average information used by the productions of a grammar.

1. Introduction

The entropy of a system is widely recognized as a measure (actually, a reciprocal measure) of that system's organization and structure [Shannon, Brillouin, Hamming, McKay, Cherry]. This suggests that the entropy of a language might be an important property to measure to form a basis for the quantitative comparison of languages. For this reason we have developed means for computing the entropies of languages. Specifically, we derive formulas for computing the entropy of a language from a grammar for that language that has been annotated with the probabilities of its productions being applied. We also show techniques whereby these production probabilities can be inferred from statistical properties of strings in the language. Finally, we apply the same techniques to several related issues, such as determining the average derivation length of a grammar, and the average information consumed by a grammar during string generation. These all seem to show promise as a means for making quantitative language comparisons.

2. Entropy of a Language

2.1 Definition of Entropy

Suppose Σ is a finite system of symbols $\sigma_1, \sigma_2, \ldots, \sigma_k$, in which symbol σ_i has a probability of occurrence p_i. Naturally, $\sum_{i=1}^{k} p_i = 1$. The *entropy* of Σ is defined

$$H(\Sigma) = \sum_{i=1}^{k} p_i \lg(1/p_i),$$

where $\lg x = \log_2 x$. Since the entropy does not depend on the symbols σ_i, and is completely determined by the probabilities p_i, it is simpler to define the entropy in terms of the *a priori* probability distribution. The entropy of the finite discrete probability distribution $p_1, p_2, \ldots, p_k$ is

$$H(p_1, p_2, \ldots, p_k) = \sum_{i=1}^{k} p_i \lg(1/p_i) = -\sum_{i=1}^{k} p_i \lg p_i.$$

The preceding ideas are easily extended to infinite discrete probability distributions. Suppose $\sum_{i=1}^{\infty} p_i = 1$. We define the entropy of this distribution:

$$H(p_1, p_2, \ldots) = \sum_{i=1}^{\infty} p_i \lg(1/p_i) = -\sum_{i=1}^{\infty} p_i \lg p_i.$$

Note that $\sum_i p_i = 1$ does not guarantee the convergence of $\sum_i p_i \lg p_i$. That is, there are probability distributions that do not have an entropy. Take, for example, $p_i = C/(i \ln^2 i)$. The sum $\sum_i p_i$ converges, but the entropy $\sum_i p_i \lg p_i$ does not. Fortunately, these troublesome distributions do not seem to occur in practice.

Entropy is widely recognized as a measure of disorganization, and thus *lack* of structure [Brillouin]. When organization increases, entropy decreases; when entropy increases, structure decreases. Thus it is usually more convenient to work with *negentropy* rather than entropy. The negentropy of a system is simply the negative of its entropy. Thus, when organization increases, so does negentropy; when negentropy

decreases, so does structure. The negentropy $\overline{H}$ of a discrete distribution p_i is defined

$$\overline{H}(p_i) = -H(p_i) = \sum_i p_i \lg p_i.$$

2.2 The Entropy of a Language

A language Σ is a (usually infinite) set of strings

$$\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_i, \ldots\}.$$

Now, let $P(\sigma_i)$ be the *a priori* probability of occurrence of the string σ_i in Σ. The negentropy of the language Σ is simply

$$\overline{H}(\Sigma) = \sum_{\sigma_i \in \Sigma} P(\sigma_i) \lg P(\sigma_i).$$

In most interesting cases the number of strings in a language is infinite. The entropy of an infinite language is thus defined in terms of an infinite set of probabilities. Therefore for most languages we are able to calculate the entropy only when there is some finite description for that infinite set of probabilities, that is, when there is some structure in that infinite set of probabilities.

Although useful languages are usually infinite (i.e., comprise an infinite number of strings), they can be described finitely by a *grammar*. That is, the grammar reflects the finite structure in the infinite set of strings. This suggests a solution to the problem of finding a finite description of the infinity of probabilities associated with the strings in the language.

The generation of each string in a language requires a finite number of elementary choices to be made. For example, in a grammar for arithmetic expressions there might be two productions for a nonterminal ν:

$\nu \rightarrow +$

$\nu \rightarrow -$

In deriving a string from this grammar, the symbol 'ν' can be replaced by either '+' or '-'; a choice must be made. Thus, a finite sequence of choices $\pi_1, \pi_2, \ldots, \pi_k$ are

necessary to determine each string in the language. Conversely, if the grammar is unambiguous, there is a unique such sequence for each string in the language[1].

Now suppose that each elementary choice π_i has an *a priori* probability $P(\pi_i)$ of being made. If these probabilities are independent, then the probability of the resulting string being generated is

$$P(\pi_1)\, P(\pi_2) \cdots P(\pi_s).$$

Thus, associating a probability with each elementary choice permitted by the grammar induces a probability on each string generated by the grammar. We call a grammar with such associated probabilities an *annotated grammar*.

There is of course no guarantee that the probabilities induced by an annotated grammar are in fact the *a priori* occurrence probabilities of the strings in the language. Indeed, an annotated grammar is a model of the processes that in reality generate strings in the language. As such, it might or might not be a good model.

We say that an annotated grammar *predicts* a language if it generates that language and induces on its strings their actual *a priori* probabilities of occurrence. We call a language *predictable* if there is an annotated grammar that predicts it. Clearly then, we can determine the probabilities of the strings in a predictable language if we can find an annotated grammar that predicts that language. Further, if we can calculate the entropy of the language generated by an annotated grammar, then we will be able to calculate the entropy of the predictable language. In the following sections we develop means for computing entropies from annotated grammars.

3. Computing Negentropy from Grammars

3.1 Annotated Regular Grammars

We begin our analysis with a particularly simple class of languages: *regular languages* [Ginsburg, Hopcroft & Ullman]. The advantage of beginning with them is that the

1. More precisely, in an unambiguous grammar there is a unique *leftmost derivation* for each string.

grammar for a regular language can be written as a single nonrecursive production making use of only a few simple operators. These operators are:

name	notation	interpretation
catenation	AB	an A followed by a B
alternation	$A \oplus B$	an A or a B
Kleene star	A^*	zero or more As
Kleene cross	A^+	one or more As

Any regular language can be described by an expression formed from the empty string (ε), individual tokens and these operators, appropriately parenthesized[2]. Such an expression, which defines a regular language, is called a *regular expression*.

For example, the regular language of signed, nonnull digit strings is defined by the regular expression:

$$(+ \oplus - \oplus \varepsilon) (0 \oplus 1 \oplus 2 \oplus 3 \oplus 4 \oplus 5 \oplus 6 \oplus 7 \oplus 8 \oplus 9)^+$$

This expression can be read, "a plus or a minus or nothing, followed by a string of one or more digits."

As discussed in Section 2, to compute the entropy of a language it is necessary to know the probabilities of the strings in that language. If we have an annotated grammar that predicts the language that it generates, then the probabilities of these strings can be computed from the production probabilities (choice probabilities) in the grammar.

In deriving a string from a regular expression there is only one situation in which a choice can be made: from $A \oplus B$ we can derive either an A or a B. Thus we can annotate a regular expression by associating probabilities with all the alternands of an alternation. We write the probabilities immediately preceding the alternands that they

2. We have used '$\oplus$' instead of the usual '|', since the latter could be confused with conditional probabilities, conditional entropies, etc. Readers unfamiliar with the regular languages and other concepts from formal language theory should consult any standard text on the subject (e.g., Ginsburg or Hopcroft & Ullman).

are associated with:

$$p\ A\ \oplus\ \bar{p}\ B.$$

This means that we can choose an A with probability p, or a B with probability $\bar{p}$. (Here, and throughout this paper, we use $\bar{p}$ as an abbreviation for $1-p$.)

Since one of the alternands *must* be chosen, their probabilities must add to unity. This is the case above, since $p+\bar{p} = 1$. It also applies if there are more that two alternands. For example, if we have

$$p_1 A_1 \oplus p_2 A_2 \oplus \quad \cdot \quad \oplus p_n A_n$$

then we must have $p_1+p_2+\cdots+p_n = 1$.

The following sections develop entropy formulas that can be recursively applied to any annotated regular expression to compute the entropy of the regular language predicted by that expression. When these results have been obtained we will show that they can be easily extended to the computation of the entropy of any context-free language from a grammar that predicts it.

3.2 Entropy Formulas

We derive a series of formulas that can be applied recursively to an annotated regular expression to compute the entropy of the language predicted by that expression. In all cases we assume that the regular expression is an unambiguous grammar (i.e., there is only one way to generate a given string), and that the choices leading to a given string are independent.

We begin with the simplest regular expressions, the empty string and individual tokens, and proceed to the catenation and alternation operations.

Theorem: If ε is the set containing just the empty string and τ the set containing just the individual symbol τ then

$$\overline{H}(\varepsilon) = \overline{H}(\tau) = 0.$$

Proof: By definition $L(\varepsilon) = \{\varepsilon\}$ and $L(\tau) = \{\tau\}$. Since there is only one symbol in each of these languages, its *a priori* probability is 1. Hence,

$$\overline{H}(\varepsilon) = \overline{H}(\tau) = 1 \lg 1 = 0.$$

Theorem: $\overline{H}(AB) = \overline{H}(A) + \overline{H}(B)$.

Proof: Suppose

$L(A) = \{\alpha_1, \alpha_2, \ldots\}$,

$L(B) = \{\beta_1, \beta_2, \ldots\}$.

Then

$L(AB) = \{\alpha_i\beta_j \mid \alpha_i \in L(A), \beta_j \in L(B)\}$.

Let $P_A(\alpha_i)$ be the probability of choosing α_i from $L(A)$ and $P_B(\beta_j)$ the probability of choosing β_j from $L(B)$, then, since we are assuming these choices are independent, the probability of choosing $\alpha_i\beta_j$ from $L(AB)$, $P_{AB}(\alpha_i\beta_j)$, is just $P_A(\alpha_i)P_B(\beta_j)$. Now, let $p_i = P_A(\alpha_i)$ and $q_i = P_B(\beta_i)$. Then, by factoring and distributing:

$$\begin{aligned}
\overline{H}(AB) &= \sum_{i,j} P_{AB}(\alpha_i\beta_j) \lg P_{AB}(\alpha_i\beta_j) \\
&= \sum_{i,j} P_A(\alpha_i)P_B(\beta_j) \lg[P_A(\alpha_i)P_B(\beta_j)] \\
&= \sum_i \sum_j p_i q_j \lg(p_i q_j) \\
&= \sum_i p_i [\sum_j q_j \lg(p_i q_j)] \\
&= \sum_i p_i [\sum_j q_j (\lg p_i + \lg q_j)] \\
&= \sum_i p_i [\sum_j q_j \lg p_i + \sum_j q_j \lg q_j].
\end{aligned}$$

Now, since $\sum_j q_j = 1$ and $\overline{H}(B) = \sum_j q_j \lg q_j$,

$$\begin{aligned}
\overline{H}(AB) &= \sum_i p_i [\lg p_i + \overline{H}(B)] \\
&= \sum_i p_i \lg p_i + \sum_i p_i \overline{H}(B).
\end{aligned}$$

Since $\sum_i p_i = 1$ and $\bar{H}(A) = \sum_i p_i \lg p_i$, we have

$$\bar{H}(AB) = \bar{H}(A) + \bar{H}(B).$$

Q.E.D

Definition: The n-fold catenation of A with itself, A^n, is defined:

$A^0 = \varepsilon$,

$A^1 = A$,

$A^{n+1} = AA^n$, for $n>0$.

Corollary: $\bar{H}(A^n) = n\bar{H}(A)$.

Proof: We prove the result inductively.

$$\bar{H}(A^0) = \bar{H}(\varepsilon) = 0 = 0 \cdot \bar{H}(A).$$

Similarly,

$$\bar{H}(A^1) = \bar{H}(A) = 1 \cdot \bar{H}(A).$$

Proceeding inductively for $n>0$,

$$\begin{aligned} \bar{H}(A^{n+1}) &= \bar{H}(AA^n) \\ &= \bar{H}(A) + \bar{H}(A^n) \\ &= \bar{H}(A) + n\bar{H}(A) \\ &= (n+1)\bar{H}(A). \end{aligned}$$

Q.E.D.

Theorem: $\bar{H}\{pA \oplus \bar{p}B\} = \bar{H}(p,\bar{p}) + p\bar{H}(A) + \bar{p}\bar{H}(B)$.

Proof: Let $G = pA \oplus \bar{p}B$. Then, to generate a string in $L(G)$ we must make a choice; with probability p we pick a string from $L(A)$, with probability $\bar{p}$ we pick a string from $L(B)$. Let σ be a string in $L(G)$. Since we are only considering unambiguous grammars, σ must have come from either $L(A)$ or $L(B)$. Suppose that $\sigma \in L(A)$. Since the probability of a selection from $L(A)$ is p, and the probability of getting σ when a selec-

tion is made from $L(A)$ is $P_A(\sigma)$, the probability $P_G(\sigma)$ of selecting σ from $L(G)$ is $pP_A(\sigma)$. Similarly, if $\sigma \in L(B)$ then $P_G(\sigma) = \bar{p}P_B(\sigma)$. These observations allow us to compute the entropy of G.

$$\begin{aligned}
\overline{H}(G) &= \sum_{\sigma \in L(G)} P_G(\sigma) \lg P_G(\sigma) \\
&= \sum_{\sigma \in L(A)} P_G(\sigma) \lg P_G(\sigma) + \sum_{\sigma \in L(B)} P_G(\sigma) \lg P_G(\sigma) \\
&= \sum_i P_G(\alpha_i) \lg P_G(\alpha_i) + \sum_j P_G(\beta_j) \lg P_G(\beta_j) \\
&= \sum_i pP_A(\alpha_i) \lg pP_A(\alpha_i) + \sum_i \bar{p}P_B(\beta_j) \lg \bar{p}P_B(\beta_j) \\
&= \sum_i pp_i \lg pp_i + \sum_j \bar{p}q_j \lg \bar{p}q_j \\
&= p\sum_i p_i \lg pp_i + \bar{p}\sum_j q_j \lg \bar{p}q_j \\
&= p\sum_i p_i (\lg p + \lg p_i) + \bar{p}\sum_j q_j (\lg \bar{p} + \lg q_j) \\
&= p[\sum_i p_i \lg p + \sum_i p_i \lg p_i] + \bar{p}[\sum_j q_j \lg \bar{p} + \sum_j q_j \lg q_j].
\end{aligned}$$

From the definitions of $\overline{H}(A)$ and $\overline{H}(B)$ and the fact that the p_i and q_i sum to 1 we get

$$\begin{aligned}
\overline{H}(G) &= p[\lg p + \overline{H}(A)] + \bar{p}[\lg \bar{p} + \overline{H}(B)] \\
&= p \lg p + \bar{p} \lg \bar{p} + p\overline{H}(A) + \bar{p}\overline{H}(B) \\
&= \overline{H}(p,\bar{p}) + \overline{H}(A) + \overline{H}(B).
\end{aligned}$$

Q.E.D.

This result is easy to generalize to the n-fold alternation:

Theorem: The negentropy of an n-fold alternation can be computed:

$$\overline{H}\{p_1A_1 \oplus p_2A_2 \oplus \cdots \oplus p_nA_n\} = \overline{H}(p_1,p_2,\ldots,p_n) + \sum_{j=1}^{n} p_j \overline{H}(A_j).$$

Proof: Let $G = p_1A_1 \oplus p_2A_2 \oplus \cdots \oplus p_nA_n$. For each $\alpha_{i,j} \in L(A_j)$ let $q_{i,j} = P_{A_j}(\alpha_{i,j})$. The proof is a simple generalization of the previous:

$$\begin{aligned}
\overline{H}(G) &= \sum_{\sigma \in L(G)} P_G(\sigma) \lg P_G(\sigma) \\
&= \sum_{j=1}^{n} \sum_{\sigma \in L(A_j)} P_G(\sigma) \lg P_G(\sigma) \\
&= \sum_{j=1}^{n} \sum_{i} p_j q_{i,j} \lg p_j q_{i,j} \\
&= \sum_{j=1}^{n} p_j [\sum_{i} q_{i,j} \lg p_j q_{i,j}] \\
&= \sum_{j=1}^{n} p_j [\sum_{i} q_{i,j} (\lg p_j + \lg q_{i,j})] \\
&= \sum_{j=1}^{n} p_j [\sum_{i} q_{i,j} \lg p_j + \sum_{i} q_{i,j} \lg q_{i,j}].
\end{aligned}$$

Since $\sum_{i} q_{i,j} = 1$, $\overline{H}(p_1, p_2, \ldots, p_n) = \sum_{j=1}^{n} p_j \lg p_j$ and $\overline{H}(A_j) = \sum_{i} q_{i,j} \lg q_{i,j}$,

$$\begin{aligned}
\overline{H}(G) &= \sum_{j=1}^{n} p_j [\lg p_j + \overline{H}(A_j)] \\
&= \sum_{j=1}^{n} p_j \lg p_j + \sum_{j=1}^{n} p_j \overline{H}(A_j) \\
&= \overline{H}(p_1, p_2, \ldots, p_n) + \sum_{j=1}^{n} p_j \overline{H}(A_j).
\end{aligned}$$

Q.E.D.

The $\oplus$ operation is associative, that is,

$$A \oplus (B \oplus C) = A \oplus B \oplus C = (A \oplus B) \oplus C.$$

We would expect the annotated version of this operation to also be associative:

$$pA \oplus \bar{p}(qB \oplus \bar{q}C) = pA \oplus \bar{p}qB \oplus \bar{p}\bar{q}C.$$

Thus, if our negentropy formula is correct, we should get the same value for the negentropy of each of these regular expressions.

Theorem: $\overline{H}\{pA \oplus \bar{p}(qB \oplus \bar{q}C)\} = \overline{H}\{pA \oplus \bar{p}qB \oplus \bar{p}\bar{q}C\}$.

Proof: We derive the negentropy of the right-hand expression:

$$\overline{H}\{pA \oplus \bar{p}qB \oplus \bar{p}\bar{q}C\} = \overline{H}(p, \bar{p}q, \bar{p}\bar{q}) + p\overline{H}(A) + \bar{p}q\overline{H}(B) + \bar{p}\bar{q}\overline{H}(C).$$

Next we derive the negentropy of the left-hand side and show it equals the expression above:

$$\begin{aligned}
&\overline{H}(pA \oplus \bar{p}(qB \oplus \bar{q}C)) \\
&= \overline{H}(p,\bar{p}) + p\overline{H}(A) + \bar{p}\overline{H}(qB \oplus \bar{q}C) \\
&= \overline{H}(p,\bar{p}) + p\overline{H}(A) + \bar{p}[\overline{H}(q,\bar{q}) + q\overline{H}(B) + \bar{q}\overline{H}(C)] \\
&= \overline{H}(p,\bar{p}) + p\overline{H}(A) + \bar{p}\overline{H}(q,\bar{q}) + \bar{p}q\overline{H}(B) + \bar{p}\bar{q}\overline{H}(C) \\
&= \overline{H}(p,\bar{p}) + \bar{p}\overline{H}(q,\bar{q}) + p\overline{H}(A) + \bar{p}q\overline{H}(B) + \bar{p}\bar{q}\overline{H}(C).
\end{aligned}$$

Thus it remains to show that

$$\overline{H}(p,\bar{p}q,\bar{p}\bar{q}) = \overline{H}(p,\bar{p}) + \bar{p}\overline{H}(q,\bar{q}).$$

Expanding the right-hand side above, rearranging, and recalling that $q + \bar{q} = 1$, we get

$$\begin{aligned}
&\overline{H}(p,\bar{p}) + \bar{p}\overline{H}(q,\bar{q}) \\
&= p\lg p + \bar{p}\lg\bar{p} + \bar{p}q\lg q + \bar{p}\bar{q}\lg\bar{q} \\
&= p\lg p + \bar{p}(q+\bar{q})\lg\bar{p} + \bar{p}q\lg q + \bar{p}\bar{q}\lg\bar{q} \\
&= p\lg p + \bar{p}q\lg\bar{p} + \bar{p}\bar{q}\lg\bar{p} + \bar{p}q\lg q + \bar{p}\bar{q}\lg\bar{q} \\
&= p\lg p + \bar{p}q(\lg\bar{p} + \lg q) + \bar{p}\bar{q}(\lg\bar{p} + \lg\bar{q}) \\
&= p\lg p + \bar{p}q\lg\bar{p}q + \bar{p}\bar{q}\lg\bar{p}\bar{q} \\
&= \overline{H}(p,\bar{p}q,\bar{p}\bar{q}).
\end{aligned}$$

Q.E.D.

We now consider the iterative constructs in regular grammars. The Kleene cross, A^+, means one or more As. Thus A^+ can be expanded as the infinite alternation:

$$A^+ = A \oplus A^2 \oplus A^3 \oplus \cdots.$$

It can also be defined by the recursive formula:

$$A^+ = A \oplus AA^+.$$

This kind of regular grammar is converted to an annotated grammar by adding a *continuation probability* p:

$$A^{p+} = \bar{p}A \oplus p\bar{p}A^2 \oplus p^2\bar{p}A^3 \oplus \cdots$$

or in its recursive form

$$A^{p+} = \bar{p}A \oplus pAA^{p+}.$$

We will derive the negentropy formula two ways, using both the infinite alternation and recursive definitions, and show that we get the same result.

Theorem: $\overline{H}\{A^{p+}\} = [\overline{H}(p,\bar{p}) + \overline{H}(A)]/\bar{p}$.

Proof: First we use the recursive formulation:

$$A^{p+} = \bar{p}A \oplus pAA^{p+}.$$

Taking the negentropy of both sides we have:

$$\begin{aligned}
\overline{H}\{A^{p+}\} &= \overline{H}\{\bar{p}A \oplus pAA^{p+}\} \\
&= \overline{H}(p,\bar{p}) + \bar{p}\overline{H}(A) + p\overline{H}\{AA^{p+}\} \\
&= \overline{H}(p,\bar{p}) + \bar{p}\overline{H}(A) + p[\overline{H}(A) + \overline{H}\{A^{p+}\}] \\
&= \overline{H}(p,\bar{p}) + \bar{p}\overline{H}(A) + p\overline{H}(A) + p\overline{H}\{A^{p+}\} \\
&= \overline{H}(p,\bar{p}) + \overline{H}(A) + p\overline{H}\{A^{p+}\}.
\end{aligned}$$

Solving now for $\overline{H}\{A^{p+}\}$:

$$(1-p)\overline{H}\{A^{p+}\} = \overline{H}(p,\bar{p}) + \overline{H}(A).$$

Hence,

$$\overline{H}\{A^{p+}\} = [\overline{H}(p,\bar{p}) + \overline{H}(A)]/\bar{p}.$$

Q.E.D.

Next we compute the negentropy directly from the infinite expansion of the iteration:

$$\begin{aligned}
\overline{H}\{A^{p+}\} &= \overline{H}\{\bar{p}A \oplus p\bar{p}A^2 \oplus p^2\bar{p}A^3 \oplus \cdots\} \\
&= \overline{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots) + \bar{p}\overline{H}(A) + \bar{p}p\overline{H}(A^2) + \bar{p}p^2\overline{H}(A^3) + \cdots.
\end{aligned}$$

Recalling that $\overline{H}(A^n) = n\overline{H}(A)$,

$$= \bar{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots) + \bar{p}[\bar{H}(A) + 2p\bar{H}(A) + 3p^2\bar{H}(A) + \cdots]$$

$$= \bar{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots) + \bar{p}[1+2p+3p^2+\cdots]\bar{H}(A).$$

Now note that if $|p|<1$ the power series expansion of $1/\bar{p}^2$ is

$$1/\bar{p}^2 = \frac{1}{(1-p)^2} = 1 + 2p + 3p^2 + \cdots.$$

Therefore

$$\bar{H}\{A^{p+}\} = \bar{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots) + \bar{p}(1/\bar{p}^2)\bar{H}(A)$$

$$= \bar{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots) + \bar{H}(A)/\bar{p}.$$

It remains to simplify $\bar{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots)$.

$$\bar{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots) = \bar{p}\lg\bar{p} + \bar{p}p\lg\bar{p}p + \bar{p}p^2\lg\bar{p}p^2 + \cdots$$

$$= \sum_{k=0}^{\infty}\bar{p}p^k\lg\bar{p}p^k$$

$$= \bar{p}\sum_{k=0}^{\infty}p^k\lg p^k\bar{p}$$

$$= \bar{p}\sum_{k=0}^{\infty}p^k[\lg p^k - \lg\bar{p}]$$

$$= \bar{p}[\sum_{k=0}^{\infty}p^k\lg p^k + \sum_{k=0}^{\infty}p^k\lg\bar{p}].$$

Now note that if $|p|<1$ the power series expansion of $1/\bar{p}$ is

$$1/\bar{p} = \frac{1}{1-p} = 1+p+p^2+p^3+\cdots = \sum_{k=0}^{\infty}p^k.$$

Therefore,

$$\bar{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots) = \bar{p}[\sum_{k=0}^{\infty}kp^k\lg p + (1/\bar{p})\lg\bar{p}]$$

$$= [\bar{p}\lg p\sum_{k=0}^{\infty}kp^k] + \lg\bar{p}$$

$$= [\bar{p}p\lg p\sum_{k=1}^{\infty}kp^{k-1}] + \lg\bar{p}.$$

Using again the power series expansion for $1/\bar{p}^2$ we have

$$\begin{aligned}\bar{H}(\bar{p}, \bar{p}p, \bar{p}p^2, \ldots) &= \bar{p}p \lg p(1/\bar{p}^2) + \lg \bar{p} \\ &= (p \lg p)/\bar{p} + \lg \bar{p} \\ &= (p \lg p + \bar{p} \lg \bar{p})/\bar{p} \\ &= \bar{H}(p, \bar{p})/\bar{p}.\end{aligned}$$

Therefore,

$$\bar{H}\{A^{p+}\} = \bar{H}(p, \bar{p})/\bar{p} + \bar{H}(A)/\bar{p} = [\bar{H}(p, \bar{p}) + \bar{H}(A)]/\bar{p}.$$

Q.E.D.

The Kleene star, A^*, means *zero* or more repetitions. Thus it can be defined by the infinite expansion

$$A^* = A^0 \oplus A^1 \oplus A^2 \oplus \cdots.$$

where $A^0 = \varepsilon$ and $A^1 = A$. Since the expression following the first $\oplus$ is just the definition of A^+, the above equation can be written

$$A^* = \varepsilon \oplus A^+.$$

The Kleene star can also be defined recursively:

$$A^* = \varepsilon \oplus AA^*.$$

This notation is annotated by attaching a continuation probability p to the star:

$$A^{p*} = \bar{p}\varepsilon \oplus pAA^*.$$

The following theorem defines its negentropy.

Theorem: $\bar{H}\{A^{p*}\} = [\bar{H}(p, \bar{p}) + p\bar{H}(A)]/\bar{p}$.

Proof: There are several ways to prove this result, corresponding to the alternate definitions of A^*.

(1) First we derive the negentropy of A^{p*} from the negentropy of A^{p+}. Since

$$A^{p*} = \bar{p}\varepsilon \oplus pA^{p+},$$

we can apply $\bar{H}$ to both sides:

$$\begin{aligned}
\overline{H}\{A^{p^*}\} &= \overline{H}\{\bar{p}\varepsilon \oplus pA^{p^+}\} \\
&= \overline{H}(p,\bar{p}) + \bar{p}\overline{H}(\varepsilon) + p\overline{H}\{A^{p^+}\} \\
&= \overline{H}(p,\bar{p}) + p[\overline{H}(p,\bar{p}) + \overline{H}(A)]/\bar{p} \\
&= [\bar{p}\overline{H}(p,\bar{p}) + p\overline{H}(p,\bar{p}) + \overline{H}(A)]/\bar{p} \\
&= [\overline{H}(p,\bar{p}) + p\overline{H}(A)]/\bar{p}.
\end{aligned}$$

Q.E.D.

(2) We can also compute the negentropy from the recursive equation

$$A^{p^*} = \bar{p}\varepsilon \oplus pAA^{p^*}.$$

We apply $\overline{H}$ to both sides and get

$$\begin{aligned}
\overline{H}\{A^{p^*}\} &= \overline{H}\{\bar{p}\varepsilon \oplus pAA^{p^*}\} \\
&= \overline{H}(p,\bar{p}) + \bar{p}\overline{H}(\varepsilon) + p\overline{H}\{AA^{p^*}\} \\
&= \overline{H}(p,\bar{p}) + \bar{p}\cdot 0 + p[\overline{H}(A) + \overline{H}\{A^{p^*}\}] \\
&= \overline{H}(p,\bar{p}) + p\overline{H}(A) + p\overline{H}\{A^{p^*}\}.
\end{aligned}$$

Grouping the unknowns on the left produces

$$(1-p)\overline{H}\{A^{p^*}\} = \overline{H}(p,\bar{p}) + p\overline{H}(A).$$

Recalling that $\bar{p} = 1-p$ we have

$$\overline{H}\{A^{p^*}\} = [\overline{H}(p,\bar{p}) + p\overline{H}(A)]/\bar{p}.$$

Q.E.D.

(3) Finally, we derive the negentropy formula from the infinite expansion

$$A^{p^*} = \bar{p}\varepsilon \oplus p\bar{p}A \oplus p^2\bar{p}A^2 \oplus \cdots.$$

Take the negentropy of both sides to get

$$\begin{aligned}
\overline{H}\{A^{p^*}\} &= \overline{H}\{\bar{p}\varepsilon \oplus p\bar{p}A \oplus p^2\bar{p}A^2 \oplus p^3\bar{p}A^3 \oplus \cdots\} \\
&= \overline{H}(\bar{p},p\bar{p},p^2\bar{p},p^3\bar{p},\ldots) + p\bar{p}\overline{H}(A) + p^2\bar{p}\overline{H}(A^2) + p^3\bar{p}\overline{H}(A^3) + \cdots \\
&= \overline{H}(\bar{p},p\bar{p},p^2\bar{p},p^3\bar{p},\ldots) + p\bar{p}(1 + 2p + 3p^2 + \cdots)\overline{H}(A) \\
&= \overline{H}(\bar{p},p\bar{p},p^2\bar{p},p^3\bar{p},\ldots) + (p/\bar{p})\overline{H}(A),
\end{aligned}$$

where we have used $1/\bar{p}^2 = 1 + 2p + 3p^2 + \cdots$. We have already shown in the deriva-
tion of $\bar{H}\{A^{p+}\}$ that

$$\bar{H}(\bar{p},\bar{p}p,\bar{p}p^2,\ldots) = \bar{H}(p,\bar{p})/\bar{p}.$$

Therefore we have

$$\begin{aligned}\bar{H}\{A^{p*}\} &= \bar{H}(p,\bar{p})/\bar{p} + (p/\bar{p})\bar{H}(A) \\ &= [\bar{H}(p,\bar{p}) + p\bar{H}(A)]/\bar{p}.\end{aligned}$$

Q.E.D.

We can check these results by computing the negentropy of A^{p+} based on equation:

$$A^{p+} = AA^{p*}$$

Applying $\bar{H}$ to both sides we derive

$$\begin{aligned}\bar{H}\{A^{p+}\} &= \bar{H}\{AA^{p*}\} \\ &= \bar{H}(A) + \bar{H}\{A^{p*}\} \\ &= \bar{H}(A) + [\bar{H}(p,\bar{p}) + p\bar{H}(A)]/\bar{p} \\ &= [\bar{H}(p,\bar{p}) + \bar{p}\bar{H}(A) + p\bar{H}(A)]/\bar{p} \\ &= [\bar{H}(p,\bar{p}) + \bar{H}(A)]/\bar{p},\end{aligned}$$

which checks with our previous result.

The formulas for computing the negentropy (and hence entropy) of a regular language are summarized in Table 1.

TABLE 1. Formulas for Negentropy of Regular Languages

$\bar{H}\{\varepsilon\}$	=	0
$\bar{H}\{t\}$	=	0
$\bar{H}\{AB\}$	=	$\bar{H}(A) + \bar{H}(B)$
$\bar{H}\{pA \oplus \bar{p}B\}$	=	$\bar{H}(p,\bar{p}) + p\bar{H}(A) + \bar{p}\bar{H}(B)$
$\bar{H}\{A^{p+}\}$	=	$[\bar{H}(p,\bar{p}) + \bar{H}(A)]/\bar{p}$
$\bar{H}\{A^{p*}\}$	=	$[\bar{H}(p,\bar{p}) + p\bar{H}(A)]/\bar{p}$

3.3 Examples

In this section we illustrate the application of our negentropy formulas with several simple examples. Several of these examples are based on *free languages:*

Definition: The *free language* on an alphabet T is the set of all finite strings (including the empty string) of elements of T.

Thus T^* is the free language on T. In most cases it does not matter what the alphabet T is, so we speak of the free language on n symbols. Let T_n represent any alphabet of n symbols:

$$T_n = \tau_1 \oplus \tau_2 \oplus \cdots \oplus \tau_n .$$

Then F_n, the free language on n symbols, is defined

$$F_n = T_n^* = (\tau_1 \oplus \cdots \oplus \tau_n)^* .$$

Of course, before we can compute the entropy of a language we must annotate the grammar with probabilities. Therefore, the annotated grammar for the free language on n symbols is

$$F_n = (q_1\tau_1 \oplus q_2\tau_2 \oplus \cdots \oplus q_n\tau_n)^{p^*} .$$

Theorem: The negentropy of the free language on n symbols,

$$F_n = (q_1\tau_1 \oplus \cdots \oplus q_n\tau_n)^{p^*} ,$$

is

$$\bar{H}(F_n) = [\bar{H}(p,\bar{p}) + p\bar{H}(q_1, \ldots, q_n)]/\bar{p} .$$

Proof: We simply apply the formulas from Table 1:

$$\begin{aligned}
\bar{H}(F_n) &= \bar{H}\{(q_1\tau_1 \oplus \cdots \oplus q_n\tau_n)^{p^*}\} \\
&= [\bar{H}(p,\bar{p}) + p\bar{H}\{q_1\tau_1 \oplus \cdots \oplus q_n\tau_n\}]/\bar{p} \\
&= [\bar{H}(p,\bar{p}) + p\{\bar{H}(q_1, \ldots, q_n) + q_1\bar{H}(\tau_1) + \cdots + q_n\bar{H}(\tau_n)\}]/\bar{p} \\
&= [\bar{H}(p,\bar{p}) + p\bar{H}(q_1, \ldots, q_n)]/\bar{p} .
\end{aligned}$$

Q.E.D.

The free language on one symbol τ is just the set of all strings of τs:

$$L(F_1) = \{\varepsilon, \tau, \tau\tau, \tau\tau\tau, \cdots\} .$$

The following theorem defines the negentropy of F_1

Corollary: The negentropy of the free language with continuation probability p on one symbol is:

$$\bar{H}(F_1) = \bar{H}(p,\bar{p})/\bar{p}.$$

Proof: We simply use the previous theorem with $n = 1$.

Corollary: The negentropy of the free language with continuation probability p on an alphabet of n equally likely symbols is

$$[\bar{H}(p,\bar{p}) - p\lg n]/\bar{p}.$$

Proof: To derive this simply set $q_i = 1/n$ in the negentropy formula for F_n:

$$\begin{aligned}
\bar{H}(F_n) &= [\bar{H}(p,\bar{p}) + p\bar{H}(q_1, \ldots, q_n)]/\bar{p} \\
&= [\bar{H}(p,\bar{p}) + p\bar{H}(1/n, \ldots, 1/n)]/\bar{p} \\
&= [\bar{H}(p,\bar{p}) + p\sum_{i=1}^{n}(1/n)\lg(1/n)]/\bar{p} \\
&= [\bar{H}(p,\bar{p}) + p\lg(1/n)]/\bar{p} \\
&= [\bar{H}(p,\bar{p}) - p\lg n]/\bar{p}.
\end{aligned}$$

Q.E.D.

Table 2 shows the entropies of free languages on equally likely symbols for several different continuation probabilities.

TABLE 2. Entropies of Free Languages on Equally Likely Symbols

$p \backslash n$	2	4	8	10	12	64	256
0.1	0.63	0.74	0.85	0.89	0.92	1.19	1.41
0.2	1.15	1.40	1.65	1.73	1.80	2.40	2.90
0.3	1.69	2.12	2.54	2.68	2.80	3.83	4.69
0.4	2.28	2.95	3.62	3.83	4.01	5.62	6.95
0.5	3.00	4.00	5.00	5.32	5.58	8.00	10.00
0.6	3.93	5.43	6.93	7.41	7.80	11.43	14.43
0.7	5.27	7.60	9.94	10.69	11.30	16.94	21.60
0.8	7.61	11.61	15.61	16.90	17.95	27.61	35.61
0.9	13.69	22.69	31.69	34.59	36.95	58.69	76.69

This table suggests that we consider the special case in which n is a power of two and p

is one half. This leads to:

Corollary: The negentropy of the free language with continuation probability one half and 2^k equally likely symbols is $-\lg k - 2$. Conversely, the entropy is $\lg k + 2$.

Proof: Let $p = \frac{1}{2}$ and $n = 2^k$ in the formula from the previous corollary and we have

$$\begin{aligned}
\overline{H} &= [\overline{H}(p,\overline{p}) - p \lg n]/\overline{p} \\
&= [\overline{H}(\tfrac{1}{2},\tfrac{1}{2}) - \tfrac{1}{2} \lg 2^k]/\tfrac{1}{2} \\
&= 2[\tfrac{1}{2} \lg \tfrac{1}{2} + \tfrac{1}{2} \lg \tfrac{1}{2} - k/2] \\
&= 2\lg \tfrac{1}{2} - \lg k \\
&= -2 - \lg k.
\end{aligned}$$

Q.E.D.

3.4 Computing the Entropy of a Context-Free Grammars

In this section we extend the results of the previous sections to the computation of the entropy of any context-free grammar.

As usual, we define a context-free grammar G to be a quadruple,

$$G = \langle T, N, P, \nu_0 \rangle,$$

in which T is a finite set of *terminal symbols*, N is a finite set of *nonterminal symbols*, $\nu_0 \in N$ is the *goal symbol*, and P is a finite set of productions,

$$P \subset N \times (T \cup N)^*.$$

That is, each production is a pair of the form $\langle \nu, \alpha \rangle$, in which ν is a nonterminal and α is a finite string of terminals and nonterminals. Such a production is usually written '$\nu \rightarrow \alpha$'. The *Backus-Naur form* (BNF) of a context free grammars combines all the productions for a given nonterminal into a single productions. For example, if context-free grammar contains the following productions for ν:

$\nu \rightarrow \alpha_1$

$\nu \rightarrow \alpha_2$

$$\vdots$$

$$\nu \rightarrow \alpha_n$$

then the BNF form of this grammar combines them into a single production:

$$\nu \rightarrow \alpha_1 \oplus \alpha_2 \oplus \cdots \oplus \alpha_n.$$

In the following discussion we will usually use the BNF form of grammars.

The characteristic that distinguishes context-free grammars from regular grammars is that the productions of a context-free grammar can be mutually recursive. That is, a nonterminal ν can be defined in terms of a string that is, directly or indirectly, defined in terms of ν. It is well known, however, (see Ginsburg) that each production in a BNF grammar can be considered an equation on sets of strings. If we recursively define $L(\alpha)$, the language defined by α, as follows:

$L(\varepsilon) = \{\varepsilon\}$

$L(\tau) = \{\tau\}$

$L(\alpha\beta) = L(\alpha) * L(\beta)$

where $S * T = \{\alpha\beta \mid \alpha \in S, \beta \in T\}$

$L(\alpha \oplus \beta) = L(\alpha) \cup L(\beta)$

then each production $\nu \rightarrow \alpha$ of a context-free grammar G can be transformed into a corresponding equation

$$L(\nu) = L(\alpha).$$

Let $G = \langle T, N, P, \nu_0 \rangle$ be a context-free grammar, in which

$$P = \{\nu_0 \rightarrow \alpha_0, \nu_1 \rightarrow \alpha_1, \ldots, \nu_k \rightarrow \alpha_k\}$$

is a set of productions in BNF form. Then corresponding to P is a collection of simultaneous equations on sets of strings:

$L(\nu_0) = L(\alpha_0)$

$L(\nu_1) = L(\alpha_1)$

$$L(\nu_k) = L(\alpha_k)$$

The solution to this set of equations defines the language generated by G, since $L(G) = L(\nu_0)$.

Context-free grammars can be annotated with production probabilities in the same way as regular grammars. Formally, we define an *annotated context-free grammar* G to be a quadruple $<T, N, P, \nu_0>$, in which $\nu_0 \in N$ and

$$P \subset \mathbb{R} \times N \times (T \cup N)^*.$$

Thus each production is a triple $<p,\nu,\alpha>$, p being a real number representing the probability of applying the production $\nu \rightarrow \alpha$. We impose the restriction that all the probabilities associated with the productions for a given nonterminal must sum to unity:

$$\sum_{<p_i,\nu,\alpha_i> \in P} p_i = 1, \text{ for } \nu \in N.$$

This is simpler to see in the BNF form of an annotated context-free grammar. In any production that is an alternation,

$$\nu \rightarrow p_1\alpha_1 \oplus p_2\alpha_2 \oplus \quad \cdots \quad \oplus p_n\alpha_n,$$

we must have that

$$\sum_{i=1}^{n} p_i = 1.$$

Consider an unambiguous annotated context-free grammar G and let $\Sigma = L(G)$ be the language generated by G. Let $P_G(\sigma)$ be the probability that a string σ is generated by G. We say that Σ is *predicted* by G if for every string σ, $P_\Sigma(\sigma) = P_G(\sigma)$, that is, the observed probability of occurrence of σ is the same as the probability of its generation by G. We now consider how we might compute the negentropy of Σ from G.

Consider a production $\nu \rightarrow \alpha$ in the annotated grammar; this corresponds to an equation $L(\nu) = L(\alpha)$. Since $\nu \rightarrow \alpha$, the probability of a string being generated from ν is the

same as its probability of being generated from α. Thus, $P_\nu(\sigma) = P_\alpha(\tau)$, for all $\sigma \in L(\nu) = L(\alpha)$. Thus, the negentropy of $L(\nu)$, which we can write $\bar{H}(\nu)$, is the same as the negentropy of $L(\alpha)$, which we can write $\bar{H}(\alpha)$. That is,

$$\bar{H}(\nu) = \bar{H}(\alpha).$$

It can be seen that corresponding to the BNF productions $\alpha_0 \to \alpha_1$ in P there is a set of simultaneous equations

$$\begin{aligned}
\bar{H}(\nu_0) &= \bar{H}(\alpha_0) \\
\bar{H}(\nu_1) &= \bar{H}(\alpha_1) \\
&\vdots \\
\bar{H}(\nu_n) &= \bar{H}(\alpha_n)
\end{aligned}$$

that can be solved to yield the negentropy of the language predicted by the grammar. In particular,

$$\bar{H}(\Sigma) = \bar{H}(G) = \bar{H}(\nu_0).$$

We have already made used this technique in applying the recursive definitions of of A^{p^*} and A^{p^+} to solve for their negentropy. In summary, the methods developed previously for computing the negentropies of regular languages can be extended in the obvious way to context-free languages.

4. Determining the Production Probabilities

4.1 Measurable Properties

To compute any specific entropies we need to know the probabilities of applying the productions in the appropriate grammar for the language. These can be obtained by determining measurable parameters whose values are implied by the production probabilities. That is, the measurable properties are a function of the production probabilities. The production probabilities can then be determined by (analytically or numerically) inverting this function.

What measurable properties should we use? One of the simplest is the density of occurrence of a token. Let $Occ_\tau(\sigma)$ be the number of occurrences of the symbol τ in the string σ. This is formally defined:

$Occ_\tau(\varepsilon) = 0,$

$Occ_\tau(\tau) = 1,$

$Occ_\tau(\tau') = 0,$ for $\tau \neq \tau',$

$Occ_\tau(\sigma) = Occ_\tau(\alpha) + Occ_\tau(\beta),$ for $\sigma = \alpha\beta.$

The $\Delta_\tau(G)$, then density of occurrence of τ in the language generated by G is

$$\Delta_\tau(G) = \frac{\sum_{\sigma \in L(G)} P_G(\sigma) Occ_\tau(\sigma)}{\sum_{\sigma \in L(G)} P_G(\sigma) |\sigma|}$$

where $P_G(\sigma)$ is the predicted probability of generation of σ and $|\sigma|$ is the length of σ. If G predicts $L(G)$, then $\Delta_\tau(G)$ will be the observed density of occurrence of τ in languages generated by G.

The formula for $\Delta_\tau(G)$ suggests two useful properties of a grammar: the average length of the strings it generates and the average number of occurrences of a token in a string. We let $\Lambda(G)$ be the predicted average length of the strings generated by G:

$$\Lambda(G) = \sum_{\sigma \in L(G)} P_G(\sigma) |\sigma|.$$

We let $\Phi_\tau(G)$ be the predicted frequency of occurrence of τ in the strings generated by G:

$$\Phi_\tau(G) = \sum_{\sigma \in L(G)} P_G(\sigma) Occ_\tau(\sigma).$$

It then follows that

$$\Delta_\tau(G) = \Phi_\tau(G) / \Lambda(G).$$

The goal then is to find ways to compute $\Phi_\tau(G)$ and $\Lambda(G)$ from G. This will permit us to calculate predicted values of Δ_τ which can be compared with actual measurements.

Therefore, the next two sections present means for calculating $\Lambda(G)$ and $\Phi_\tau(G)$.

4.2 Average String Length

We begin again with regular grammars.

Theorem: The average lengths of the empty and single token grammars are defined:

$\Lambda(\varepsilon) = 0$ tokens,

$\Lambda(\tau) = 1$ token.

Proof: Obvious.

For the remaining derivations we need some notation. Suppose that

$L(A) = \{\alpha_1, \alpha_2, \ldots\}$,

$L(B) = \{\beta_1, \beta_2, \ldots\}$,

$a_i = P_A(\alpha_i)$,

$b_j = P_B(\beta_j)$.

Then it follows that

$$\Lambda(A) = \sum_i a_i |\alpha_i|,$$

$$\Lambda(B) = \sum_j b_j |\beta_j|.$$

Theorem: The average length of the catenation of two grammars is the sum of their average lengths:

$$\Lambda(AB) = \Lambda(A) + \Lambda(B).$$

Proof: Note that is $\sigma \in L(AB)$ then $\sigma = \alpha_i \beta_j$ for some $\alpha_i \in L(A)$, $\beta_j \in L(B)$. Assuming as usual that the choices from A and B are independent,

$$P_{AB}(\sigma) = P_A(\alpha_i) P_B(\beta_j) = a_i b_j.$$

We now derive the average length:

$$\begin{aligned}
\Lambda(AB) &= \sum_i \sum_j P_{AB}(\alpha_i\beta_j)|\alpha_i\beta_j| \\
&= \sum_i \sum_j a_i b_j (|\alpha_i| + |\beta_j|) \\
&= \sum_i a_i [\sum_j b_j |\alpha_i| + \sum_j b_j |\beta_j|] \\
&= \sum_i a_i [|\alpha_i| + \Lambda(B)] \\
&= \sum_i a_i |\alpha_i| + \sum_i a_i \Lambda(B) \\
&= \Lambda(A) + \Lambda(B).
\end{aligned}$$

Q.E.D.

Corollary: $\Lambda(A^n) = n\Lambda(A)$

Theorem: The average length of an alternation is the average of the average lengths of the alternands:

$$\Lambda(pA \oplus \bar{p}B) = p\Lambda(A) + \bar{p}\Lambda(B).$$

Proof: Let $G = pA \oplus \bar{p}B$. Recall that if $\sigma \in L(G)$ then either $\sigma \in L(A)$ or $\sigma \in L(B)$, and that the choice from A is made with probability p. Therefore

$P_G(\sigma) = pP_A(\sigma)$, if $\sigma \in L(A)$,

$P_G(\sigma) = \bar{p}P_B(\sigma)$, if $\sigma \in L(B)$.

Then we derive:

$$\begin{aligned}
\Lambda(G) &= \sum_{\sigma \in L(G)} P_G(\sigma)|\sigma| \\
&= \sum_i P_G(\alpha_i)|\alpha_i| + \sum_j P_G(\beta_j)|\beta_j| \\
&= \sum_i p a_i |\alpha_i| + \sum_j \bar{p} b_j |\beta_j| \\
&= p\sum_i a_i |\alpha_i| + \bar{p}\sum_j b_j |\beta_j| \\
&= p\Lambda(A) + \bar{p}\Lambda(B).
\end{aligned}$$

Q.E.D.

Theorem: $\Lambda(A^{p+}) = \Lambda(A)/\bar{p}$.

Proof: We appeal to the infinite expansion:

$$A^{p+} = \bar{p}A \oplus p\bar{p}A^2 \oplus p^2\bar{p}A^3 \oplus \cdots.$$

Applying Λ to both sides:

$$\begin{aligned}\Lambda(A^{p+}) &= \bar{p}\Lambda(A) + p\bar{p}\Lambda(A^2) + p^2\bar{p}\Lambda(A^3) + \cdots \\ &= \bar{p}[1 + 2p + 3p^2 + \cdots]\Lambda(A) \\ &= \bar{p}[1/\bar{p}^2]\Lambda(A) \\ &= \Lambda(A)/\bar{p}.\end{aligned}$$

Q.E.D. Alternately, we can appeal to the recursive definition:

$$\begin{aligned}\Lambda(A^{p+}) &= \bar{p}\Lambda(A) + p\Lambda(AA^{p+}) \\ &= \bar{p}\Lambda(A) + p\Lambda(A) + p\Lambda(A^{p+}).\end{aligned}$$

Grouping like terms gives

$$(1-p)\Lambda(A^{p+}) = (\bar{p}+p)\Lambda(A).$$

which leads directly to the result. *Q.E.D.*

Theorem: $\Lambda(A^{p*}) = (p/\bar{p})\Lambda(A)$.

Proof: We apply Λ to the infinite expansion of A^{p*}:

$$\begin{aligned}\Lambda(A^{p*}) &= \bar{p}\Lambda(\varepsilon) + p\bar{p}\Lambda(A) + p^2\bar{p}\Lambda(A^2) + p^3\bar{p}\Lambda(A^3) + \cdots \\ &= \bar{p}\cdot 0 + p\bar{p}\Lambda(A) + p^2\bar{p}\cdot 2\Lambda(A) + p^3\bar{p}\cdot 3\Lambda(A) \\ &= p\bar{p}(1 + 2p + 3p^2 + \cdots)\Lambda(A) \\ &= p\bar{p}(1/\bar{p}^2)\Lambda(A) \\ &= (p/\bar{p})\Lambda(A).\end{aligned}$$

Q.E.D.

Alternately, we can apply Λ to the recursive definition:

$$\Lambda(A^{p*}) = \bar{p}\Lambda(\varepsilon) + p\Lambda(AA^{p*})$$

$$= \bar{p}\ 0 + p\Lambda(A) + p\Lambda(A^{p^*}).$$

Solving for $\Lambda(A^{p^*})$ we get

$$\Lambda(A^{p^*}) = (p/\bar{p})\Lambda(A).$$

Q.E.D.

We consider some simple examples based on free languages.

Theorem: Consider the free language on n symbols generated by:

$$F_n = (q_1\tau_1 \oplus q_2\tau_2 \oplus \cdots \oplus q_n\tau_n)^{p^*}.$$

The average length of the strings of this language is

$$\lambda = \Lambda(F_n) = p/\bar{p} \text{ tokens.}$$

Proof: Since $\Lambda(\tau_i) = 1$ and $q_1 + \cdots + q_n = 1$,

$$\begin{aligned}\Lambda(F_n) &= (p/\bar{p})[q_1\Lambda(\tau_1) + \cdots + q_n\Lambda(\tau_n)] \\ &= (p/\bar{p})[q_1 + \cdots + q_n] \\ &= p/\bar{p}.\end{aligned}$$

Q.E.D.

Notice that the average length of a free language is independent of the number of symbols in the alphabet. This is to be expected.

Corollary: The average length of a free language with continuation probability ½ is 1 token.

Proof: Apply the previous theorem with $p = \bar{p} = ½$. *Q.E.D.*

TABLE 3. Average String Length for Regular Grammars

$\Lambda(\varepsilon)$	=	0 tokens
$\Lambda(\tau)$	=	1 token
$\Lambda(AB)$	=	$\Lambda(A) + \Lambda(B)$
$\Lambda(pA \oplus \bar{p}B)$	=	$p\Lambda(A) + \bar{p}\Lambda(B)$
$\Lambda(A^{p+})$	=	$\Lambda(A)/\bar{p}$
$\Lambda(A^{p^*})$	=	$(p/\bar{p})\Lambda(A)$

The formulas for computing average string length are summarized in Table 3.

4.3 Average Token Frequency

The formulas for computing average token frequency are almost identical to those for average length. For this reason the proofs are omitted and the results are shown in Table 4.

TABLE 4. Average Token Frequency for Regular Grammars

$\Phi_\tau(\varepsilon)$	=	0
$\Phi_\tau(\tau)$	=	1
$\Phi_\tau(\tau')$	=	0, for $\tau \neq \tau'$
$\Phi_\tau(AB)$	=	$\Phi_\tau(A) + \Phi_\tau(B)$
$\Phi_\tau(pA \oplus \bar{p}B)$	=	$p\Phi_\tau(A) + \bar{p}\Phi_\tau(B)$
$\Phi_\tau(A^{p+})$	=	$\Phi_\tau(A)/\bar{p}$
$\Phi_\tau(A^{p*})$	=	$(p/\bar{p})\Phi_\tau(A)$

Theorem: Consider the free language on n symbols:

$$F_n = (q_1\tau_1 \oplus q_2\tau_2 \oplus \cdots \oplus q_n\tau_n)^{p*}.$$

The frequency of occurrence of token τ_i is

$$\varphi_i = \Phi_{\tau_i}(F_n) = q_i p/\bar{p}.$$

Proof: We derive as follows, abbreviating Φ_{τ_i} by Φ_i:

$$\begin{aligned}
\Phi_i(F_n) &= (p/\bar{p})\Phi_i[q_1\tau_1 \oplus \cdots \oplus q_n\tau_n] \\
&= (p/\bar{p})[q_1\Phi_i(\tau_1) + \cdots + q_i\Phi_i(\tau_i) + \cdots + q_n\Phi_i(\tau_n)] \\
&= (p/\bar{p})[q_1 \cdot 0 + \cdots + q_i \cdot 1 + \cdots + q_n \cdot 0] \\
&= (p/\bar{p})q_i.
\end{aligned}$$

Q.E.D.

Corollary: In the free language of the previous theorem, the density of occurrence of symbol τ_i is q_i. We denote this measurable property δ_i.

Proof: Since $\Delta_i(F_n) = \Phi_i(F_n)/\Lambda(F_n)$, we have

$$\begin{aligned}
\delta_i &= \varphi_i/\lambda \\
&= \frac{q_i p/\bar{p}}{p/\bar{p}} \\
&= q_i.
\end{aligned}$$

Q.E.D.

The following corollary shows that for the case of a free language it is easy to compute the production probabilities from measurable properties of strings in the language.

Corollary: If we measure the properties δ_1, δ_2, ..., δ_n and λ of a free language on n symbols, then we can compute the probabilities q_1, q_2, ..., q_n and p from them by the formulas:

$$q_i = \delta_i,$$

$$p = \lambda/(\lambda+1).$$

Proof: The formulas for q_i are obvious. For p we know that

$$\lambda = p/\bar{p} = p/(1-p)$$

Therefore $\lambda - p\lambda = p$, so $\lambda = p + p\lambda$. Thus $\lambda = p(\lambda+1)$, so $p = \lambda/(\lambda+1)$.

Q.E.D.

Corollary: The negentropy of a free language exhibiting occurrence densities δ_1, ..., δ_n and average string length λ is:

$$\bar{H}_n = \bar{H}(\lambda, \lambda+1) + \lambda\bar{H}(\delta_1, \ldots, \delta_n).$$

Proof: To derive this result we take the formula for the negentropy of a free language,

$$\bar{H}_n = \bar{H}(F_n) = [\bar{H}(p,\bar{p}) + p\bar{H}(q_1, \ldots, q_n)]/\bar{p},$$

and substitute the values for q_i and p derived in the previous corollary. To do this, note that

$$\bar{p} = 1-p = 1 - \frac{\lambda}{\lambda+1} = \frac{1}{\lambda+1}.$$

Then we have

$$\overline{H}_n = \frac{\overline{H}\left(\frac{\lambda}{\lambda+1}, \frac{1}{\lambda+1}\right) + \frac{\lambda}{\lambda+1}\overline{H}(\delta_1, \ldots, \delta_n)}{\frac{1}{\lambda+1}}$$

$$= (\lambda+1)\left[\frac{\lambda}{\lambda+1}\lg\frac{\lambda}{\lambda+1} + \frac{1}{\lambda+1}\lg\frac{1}{\lambda+1} + \frac{\lambda}{\lambda+1}\overline{H}(\delta_1, \ldots, \delta_n)\right]$$

$$= \lambda \lg\frac{\lambda}{\lambda+1} - \lg(\lambda+1) + \lambda\overline{H}(\delta_1, \ldots, \delta_n)$$

$$= \lambda \lg\lambda - \lambda\lg(\lambda+1) - \lg(\lambda+1) + \lambda\overline{H}(\delta_1, \ldots, \delta_n)$$

$$= \lambda \lg\lambda - (\lambda+1)\lg(\lambda+1) + \lambda\overline{H}(\delta_1, \ldots, \delta_n)$$

$$= \overline{H}(\lambda, \lambda+1) + \lambda\overline{H}(\delta_1, \ldots, \delta_n).$$

Q.E.D.

Thus we have the negentropy (and hence entropy) of a language expressed entirely in measurable parameters.

4.4 Average Information per Symbol

Recall that the entropy of a language measures the average information born by each *string* in the language. That is,

$$H(\Sigma) = \sum_{\sigma\in\Sigma} P_L(\sigma) I_\Sigma(\sigma).$$

However, each of the strings of the language is composed of a number of terminal symbols (tokens). Therefore, it is interesting to compute the average information born by each *symbol* (token) in the strings in the language. We call this the *information*

density of the language.

Information density is easy to compute: it is simply the average information born by the strings of the language divided by the average length of those strings:

$$\eta(\Sigma) = H(\Sigma)/\Lambda(\Sigma),$$

where we have used $\eta(\Sigma)$ for the average information born per symbol in the strings Σ. The units of information density are bits/token.

If the grammar G predicts the language Σ, then

$$\eta(\Sigma) = \eta(G) = H(G)/\Lambda(G).$$

We use this result to compute the information density for several languages.

Theorem: Let F_n be the free language with continuation probability p on n symbols with probabilities q_i. Then, the information density of F_n is:

$$\eta(F_n) = H(p,\bar{p})/p + H(q_1, \ldots, q_n) \quad \text{bits/token.}$$

Proof: Take the formula for the negentropy of F_n and negate it to get the entropy formula:

$$H(F_n) = [H(p,\bar{p}) + pH(q_1, \ldots, q_n)]/\bar{p}.$$

Divide this by the average length $\Lambda(F_n) = p/\bar{p}$ to get the information density:

$$\eta(F_n) = H(F_n)/\Lambda(F_n)$$

$$= \frac{[H(p,\bar{p}) + pH(q_1, \ldots, q_n)]/\bar{p}}{p/\bar{p}}$$

$$= H(p,\bar{p})/p + H(q_1, \ldots, q_n).$$

Q.E.D.

Corollary: The information density of the free language on one symbol with continuation probability p is:

$$\eta(F_1) = H(p,\bar{p})/p \quad \text{bits/token.}$$

Corollary: The information density of the free language with continuation probability p on n equally likely symbols is:

$$\eta = H(p,\bar{p})/p + \lg n \quad \text{bits/token.}$$

Proof: We simply use $q_i = 1/n$:

$$\begin{aligned}\eta &= H(p,\bar{p})/p + H(q_1, \ldots, q_n) \\ &= H(p,\bar{p})/p + H(1/n, \ldots, 1/n) \\ &= H(p,\bar{p})/p + (1/n)\lg n + \cdots + (1/n)\lg n \\ &= H(p,\bar{p})/p + \lg n \quad \text{bits/token.}\end{aligned}$$

Q.E.D.

Corollary: The information density of the free language with continuation probability one half and 2^k equally likely symbols is $k+2$ bits/token.

Proof: Let $n = 2^k$ and $p = \bar{p} = \frac{1}{2}$ in the previous formula:

$$\begin{aligned}\eta &= H(\tfrac{1}{2},\tfrac{1}{2})/(\tfrac{1}{2}) + \lg 2^k \\ &= 2H(\tfrac{1}{2},\tfrac{1}{2}) + k \\ &= 2(\tfrac{1}{2}\lg 2 + \tfrac{1}{2}\lg 2) + k \\ &= 2(\tfrac{1}{2} + \tfrac{1}{2}) + k.\end{aligned}$$

Q.E.D.

The difference between the entropy of a language and its average information density can be understood by looking at some simple examples. In particular we will consider the languages N_k of all nonempty strings on k symbols. Thus, N_k is just the free language F_k without the empty string. Conversely,

$$F_k = N_k \oplus \varepsilon.$$

Theorem: Let $N_k = (q_1\tau_1 \oplus \cdots \oplus q_k\tau_k)^{p+}$. Then,

$$\begin{aligned}\overline{H}(N_k) &= [\overline{H}(p,\bar{p}) + \overline{H}(q_1, \ldots, q_k)]/\bar{p} \quad \text{bits} \\ \Lambda(N_k) &= 1/\bar{p} \quad \text{tokens}\end{aligned}$$

$$\eta(N_k) = \bar{H}(p,\bar{p}) + \bar{H}(q_1, \ldots, q_k) \text{ bits/token.}$$

Proof: Simply apply the previous formulas. *Q.E.D.*

To understand the implications of this result we consider an especially simple case, N_1, the language of all nonempty strings on one symbol τ:

$$N_1 = \tau^{P+}.$$

Hence $L(N_1) = \{\tau, \tau\tau, \tau\tau\tau, \ldots\}$.

Corollary: If the continuation probability of N_1 is one half, then

$\bar{H}(N_1) = 2$ bits

$\lambda(N_1) = 2$ tokens

$\eta(N_1) = 1$ bit/token

Proof: Substitute $\bar{p} = \frac{1}{2}$ in the previous formulas and recall that

$$H(\tfrac{1}{2},\tfrac{1}{2}) = \tfrac{1}{2}\lg 2 + \tfrac{1}{2}\lg 2 = \lg 2 = 1 \text{ bit.}$$

Q.E.D.

This result is easy to interpret, since each succeeding token indicates that the choice has been made to continue the string. Since the probability of the choice is one half, each token conveys one bit of information.

Next we consider N_2, which can be considered the language of nonnull strings of binary digits:

$$N_2 = (0 \oplus 1)^+.$$

The following theorem addresses the information density of this language.

Theorem: If N_2 is the language of nonnull binary strings:

$$N_2 = (q\,0 \oplus \bar{q}\,1)^{P+},$$

then

$\bar{H}(N_2) = [\bar{H}(p,\bar{p}) + \bar{H}(q,\bar{q})]/\bar{p}$ bits

$\Lambda(N_2) = 1/\bar{p}$ tokens

$\eta(N_2) = H(p,\bar{p}) + H(q,\bar{q})$ bits/token.

Proof: Apply the previous formulas. *Q.E.D.*

Corollary: Suppose the binary digits 0 and 1 are equally likely. Then the information density of the language of nonnull binary strings is

$$\eta(N_2) = 1 + H(p,\bar{p}) \text{ bits/token.}$$

Proof: Apply the previous theorem with $q = \bar{q} = \frac{1}{2}$. *Q.E.D.*

Note that since $H(p,\bar{p})>0$, we know

$$\eta(N_2) > 1 \text{ bit/token.}$$

Since in this case a token is a binary digit, we have the somewhat surprising result that the information density of the language of binary strings is greater than one bit per binary digit. How can this be? The extra $H(p,\bar{p})$ bits of information per binary digit comes from the fact that the binary strings are variable length. We previously saw that in N_1 the continuation of the string conveys $H(p,\bar{p})$ bits of information.

The source of the extra information can be made clearer by considering a language in which it's absent, the language of all n digit binary strings:

$$W_n = (q\,0 \oplus \bar{q}\,1)^n.$$

Let $q = \bar{q} = \frac{1}{2}$. Then

$H(W_n) = nH(\frac{1}{2},\frac{1}{2}) = n$ bits

$\Lambda(W_n) = n\,\Lambda\{q\,0 \oplus \bar{q}\,1\} = n$ tokens

$\eta(W_n) = n/n = 1$ bit/token.

Thus the information density of W_n is one bit per binary digit, as expected. Since all the strings of W_n are the same length, no information is conveyed by the continuation of the string.

Then the average derivation length of F_n is

$$D(F_n) = (p+1)/\bar{p} \quad \text{productions.}$$

Proof: We transform the productions into the equations:

$$D(F_n) = 1 + \bar{p}D(\varepsilon) + pD(AF_n)$$

$$D(A) = 1 + q_1 D(\tau_1) + \cdots + q_n D(\tau_n).$$

Thus, $D(A) = 1$. Simplifying we derive

$$D(F_n) = 1 + p[D(A) + D(F_n)]$$

$$= 1 + p + pD(F_n).$$

Solving for $D(F_n)$ we have

$$D(F_n) = (p+1)/\bar{p} \quad \text{productions.}$$

Q.E.D.

As would be expected, the average derivation length is independent of the probabilities q_i.

Corollary: The average derivation length of F_n with continuation probability one half is 3 productions.

Proof: Apply theorem with $p = \bar{p} = \frac{1}{2}$. *Q.E.D.*

This result is also intuitive. We always must apply at least one production (for F_n). If we choose to stop, with probability one half, we have applied one production. However, if we choose to continue, with probability one half, then we must apply two more productions (one for A, one for F_n), and repeat our choice. Thus we have:

$$D = \tfrac{1}{2}\cdot 1 + \tfrac{1}{2}[2 + \tfrac{1}{2}\cdot 1 + \tfrac{1}{2}(2 + \tfrac{1}{2}\cdot 1 + \cdots)]$$

$$= \tfrac{1}{2} + 1 + \tfrac{1}{2}^2 + \tfrac{1}{2} + \tfrac{1}{2}^3 + \cdots$$

Regrouping gives

$$D = (1 + \tfrac{1}{2} + \cdots) + (\tfrac{1}{2} + \tfrac{1}{2}^2 + \tfrac{1}{2}^3 + \ldots)$$

= 2 + 1 productions.

5.2 Average Information Used by a Production

Consider a leftmost derivation of a terminal string from a grammar. At each point in the derivation a nonterminal must be replaced by a string according to the productions of the grammar. In many of these cases there will be a choice of which of several alternate productions are to be applied. Such a choice will require information to be supplied. Considering the average information that must be supplied per applied production gives us a gauge of the efficiency with which a grammar transforms information into terminal strings.

Since in an unambiguous context-free grammar there is a unique leftmost derivation of any string in the language generated by the grammar, we can set up a one-one correspondence between the leftmost derivations and the strings. Thus, for any $\sigma \in L(G)$ there is a unique series of productions $\pi_1, \pi_2, \ldots, \pi_k$ that generates σ. As we saw before, if production π_i has an *a priori* probability $P(\pi_i)$ of being chosen, then the probability that G will generate σ is

$$P_G(\sigma) = P(\pi_1)P(\pi_2)\cdots P(\pi_k).$$

Hence, the generation of a string by a grammar can be viewed as a series of choices $\pi_1, \ldots, \pi_k$, having probabilities $P(\pi_1), \ldots, P(\pi_k)$ of being made.

Now we look at this formula a different way. Recall [Shannon, Hamming] that when a previously undetermined situation with *a priori* probability p is determined, the information conveyed is $-\lg p$ bits. That is, information is conveyed by making choices. Thus, the information conveyed by making choice π_i, with probability $P(\pi_i)$, is

$$I(\pi_i) = -\lg P(\pi_i) \text{ bits.}$$

Therefore, the information conveyed by σ is just the total information conveyed by the choices that lead to σ:

$$I_G(\sigma) = -\lg P_G(\sigma) = -\lg P(\pi_1) + \cdots + -\lg P(\pi_k) = I(\pi_1) + \cdots + I(\pi_k).$$

This information is used by the grammar in going from an undetermined nonterminal symbol to a completely determined terminal string. That is, this information drives a decrease in the entropy from $H(G)$ to 0 (since a terminal string has no entropy).

Now recall that

$$H(G) = -\sum_{\sigma \in L(G)} P_G(\sigma) \lg P_G(\sigma)$$

$$= \sum_{\sigma \in L(G)} P_G(\sigma) I_G(\sigma).$$

Thus, the entropy of a language is the average information conveyed by its strings.

A grammar with higher entropy is less constrained, more disordered, than one with lower entropy, so on the average it takes more information to generate a particular string from it. A grammar with low entropy is highly constrained, so on the average little information is needed (or used) in generating a string; there are fewer choices to be made.

We can now apply these results to determining the average information used per production by a grammar. Since, the information conveyed by a string is the same as the information used in its derivation, the entropy of a language measures the average information used by a grammar in generating a string of that language. If we also know the average derivation length for that grammar, that is, the average number of productions needed to generate a string, then we know the average amount Q of information consumed per production. Summarizing,

$$Q(G) = H(G) / D(G),$$

where $H(G) = H[L(G)]$.

Theorem: Consider the following grammar for the free language on n symbols:

$$F_n \rightarrow \bar{p}\,\varepsilon \oplus p A F_n$$

$$A \rightarrow q_1 \tau_1 \oplus \cdots \oplus q_n \tau_n.$$

This grammar uses on the average

$$Q(F_n) = [H(p,\bar{p}) + pH(q_1, \ldots, q_n)]/(p+1) \text{ bits/production}$$

to generate a string.

Proof: Recall that

$$H(F_n) = [H(p,\bar{p}) + pH(q_1, \ldots, q_n)]/\bar{p} \text{ bits}$$

$$D(F_n) = (p+1)/\bar{p} \text{ productions,}$$

and divide.

Q.E.D.

Corollary: F_2 with continuation probability one half uses on the average one bit of information per production.

Proof: Set $p = \bar{p} = q_1 = q_2 = \frac{1}{2}$. Then,

$$\begin{aligned} Q(F_2) &= [H(\tfrac{1}{2},\tfrac{1}{2}) + \tfrac{1}{2} H(\tfrac{1}{2},\tfrac{1}{2})]/(\tfrac{1}{2} + 1) \\ &= (1\tfrac{1}{2}) H(\tfrac{1}{2},\tfrac{1}{2})/(1\tfrac{1}{2}) \\ &= H(\tfrac{1}{2},\tfrac{1}{2}) \\ &= 1 \text{ bit.} \end{aligned}$$

Q.E.D.

This is intuitive, since this grammar must use one bit on each production, either to decide whether or not to continue, or to decide which symbol to generate.

Consider F_1, the above grammar restricted to generate the free language on one symbol, with continuation probability one half. The above formula says the information consumed per production is

$$\begin{aligned} Q(F_1) &= [H(\tfrac{1}{2},\tfrac{1}{2}) + \tfrac{1}{2} H(1)]/(\tfrac{1}{2} + 1) \\ &= H(\tfrac{1}{2},\tfrac{1}{2})/(1\tfrac{1}{2}) \\ &= 2/3 \text{ bits/production.} \end{aligned}$$

This might be surprising, since it seems that with each successive symbol of the string exactly one bit of information is being used, namely, to decide whether or not to

continue. The source of the inefficiency can be found by looking at F_1:

$$F_1 \rightarrow \bar{p}\,\varepsilon \oplus AF_1$$

$$A \rightarrow \tau$$

There is a redundant production $A \rightarrow \tau$ that uses no information; this decreases the average information used per production. If we eliminate this redundancy, we get the one-production grammar

$$F_1 \rightarrow \bar{p}\,\varepsilon \oplus p\tau F_1.$$

It has the same entropy as the previous grammar, but a shorter average derivation length:

$$\begin{aligned} D(F_1) &= 1 + pD(F_1) \\ &= 1/\bar{p} \end{aligned}$$

For $p = \frac{1}{2}$ its average derivation length is 2 productions, as opposed to 3 for the version with the redundant rule. The information used per production is then

$$\begin{aligned} Q(F_1) &= H(F_1)/D(F_1) \\ &= \frac{H(p,\bar{p})/\bar{p}}{1/\bar{p}} \\ &= H(p,\bar{p}) \quad \text{bits/production.} \end{aligned}$$

In the case $p = \frac{1}{2}$ this grammar uses one bit per production, as would be expected. Thus we have a way of comparing the efficiencies with which grammars use information and of determining whether grammars have useless productions.

6. Example Applications

In this example we illustrate the previously described techniques by their application to a nontrivial language. Table 6 shows the context-free grammar for lambda-calculus expressions; we have added unknown production probabilities. We now apply $\overline{H}$ to these productions and solve for the negentropy. Thus,

TABLE 6. Annotated Grammar for Lambda Calculus

E	=	$q_1\, I$
	$\oplus$	q_2 '(λ I E)'
	$\oplus$	q_3 '(E E)'
I	=	$(p_1$a $\oplus$ p_2b $\oplus$ $\cdots$ $\oplus$ p_{36}9$)^{P+}$

where $q_1+q_2+q_3 = 1$, and $p_1+p_2+\cdots p_{36} = 1$

$$\bar{H}(E) = \bar{H}(q_1,q_2,q_3) + q_1\bar{H}(I) + q_2\bar{H}\{\text{'(λ } I\ E\text{)'}\} + q_3\bar{H}\{\text{'(}\ E\ E\text{)'}\}.$$

Tokens can be ignored in computing negentropies, so this reduces to

$$\begin{aligned}\bar{H}(E) &= \bar{H}(q_1,q_2,q_3) + q_1\bar{H}(I) + q_2[\bar{H}(I) + \bar{H}(E)] + q_3 2\bar{H}(E)\\ &= \bar{H}(q_1,q_2,q_3) + (q_1+q_2)\bar{H}(I) + (q_2 + 2q_3)\bar{H}(E).\end{aligned}$$

Solving for $\bar{H}(E)$ we have

$$\bar{H}(E) = \frac{\bar{H}(q_1,q_2,q_3) + (q_1+q_2)\bar{H}(I)}{1 - q_2 - 2q_3}.$$

It remains to solve for $\bar{H}(I)$. We apply the formula for $\bar{H}(A^{P+})$ to get

$$\begin{aligned}\bar{H}(I) &= \bar{H}\{(p_1\text{a} \oplus \cdots\ p_{36}9)^{P+}\}\\ &= [\bar{H}(p,\bar{p}) + \bar{H}\{p_1\text{a} \oplus \cdots \oplus P_{36}9\}]/\bar{p}\\ &= [\bar{H}(p,\bar{p}) + \bar{H}(p_1,p_2,\ldots,p_{36})]/\bar{p}.\end{aligned}$$

The resulting formula for the negentropy of the lambda-calculus is:

$$\bar{H} = \frac{\bar{H}(q_1,q_2,q_3) + (q_1+q_2)[\bar{H}(p,\bar{p}) + \bar{H}(p_1,\ldots,p_{36})]/\bar{p}}{1 - q_2 - 2q_3}\text{ bits.}$$

To actually compute the negentropy it is, of course, necessary to determine the production probabilities p, q_1, q_2, q_3, p_1, p_2, ..., p_{36}. Since all the probabilities associated in an alternation must add to unity, there are just 38 independent probabilities to be determined.

To determine these probabilities we will calculate the measurable properties of the strings in the language: the average string length and the occurrence densities of the tokens. The average length is:

$$\begin{aligned}
\lambda &= \Lambda(E) \\
&= q_1\Lambda(I) + q_2\Lambda\{(\lambda\ I\ E\ )\} + q_3\Lambda\{(E\ E\ )\} \\
&= q_1\Lambda(I) + q_2[\Lambda\{(\} + \Lambda\{\lambda\} + \Lambda(I) + \lambda + \Lambda\{)\}] + q_3[\Lambda\{(\} + 2\lambda + \Lambda\{)\}] \\
&= q_1\Lambda(I) + q_2[\Lambda(I) + \lambda + 3] + q_3[\lambda + 3] \\
&= (q_1+q_2)\Lambda(I) + (q_2+2q_3)\lambda + 3q_2 + 2q_3.
\end{aligned}$$

Solving for λ we have

$$\lambda = \frac{(q_1+q_2)\Lambda(I) + 3q_2 + 2q_3}{1 - q_2 - 2q_3}.$$

It remains to compute $\Lambda(I)$:

$$\begin{aligned}
\Lambda(I) &= \Lambda\{(p_1 a \oplus \cdots \oplus p_{36}C)^+\} \\
&= \Lambda\{p_1 a \oplus \cdots \oplus p_{36}9\}/\bar{p} \\
&= (p_1 + \cdots + p_{36})/\bar{p} \\
&= 1/\bar{p} \text{ tokens.}
\end{aligned}$$

Substituting this result into the formula for λ gives the average length of a string in the lambda-calculus:

$$\lambda = \frac{(q_1+q_2)/\bar{p} + 3q_2 + 2q_3}{1 - q_2 - 2q_3} \text{tokens.}$$

Recalling that the information density of a language is the ratio of its entropy and average length, we have

$$\eta = \frac{H(q_1,q_2,q_3) + (q_1+q_2)[H(p,\bar{p}) + H(p_1,\ldots,p_{36})]/\bar{p}}{(q_1+q_2)/\bar{p} + 3q_2 + 2q_3} \text{ bits/token.}$$

It remains to compute the frequencies of occurrence of the tokens in the language. First we compute φ_{lp}, the average number of left parentheses in a string.

$$\begin{aligned}
\varphi_{lp} &= \Phi_{lp}(e) \\
&= q_1\Phi_{lp}(I) + q_2\Phi_{lp}\{(\ (\ \lambda\ I\ E\ )\ )\} + q_3\Phi_{lp}\{(\ (\ E\ E\ )\ )\} \\
&= q_1\cdot 0 + q_2(1+\varphi_{lp}) + q_3(1+\varphi_{lp}+\varphi_{lp}) \\
&= q_2 + q_3 + (q_2 + 2q_3)\varphi_{lp}.
\end{aligned}$$

Solving for φ_{lp} we have

$$\varphi_{lp} = \frac{q_2 + q_3}{1 - q_2 - 2q_3}.$$

Clearly $\varphi_{lp} = \varphi_{rp}$. It is also easy to compute the corresponding digraph rates:

$$\delta_{lp} = \delta_{rp} = \varphi_{lp}/\lambda = \varphi_{rp}/\lambda.$$

Next we compute φ_λ, the frequency of the token 'λ', by the same process:

$$\begin{aligned}
\varphi_\lambda &= \Phi_\lambda(e) \\
&= q_1\Phi_\lambda(I) + q_2\Phi_\lambda\{(\ \lambda\ I\ E\ )\} + q_3\Phi_\lambda\{(\ E\ E\ )\} \\
&= q_2[\Phi_\lambda(\lambda) + \Phi_\lambda(E)] + 2q_3\Phi_\lambda(E) \\
&= q_2(1 + \varphi_\lambda) + 2q_3\varphi_\lambda.
\end{aligned}$$

Solving for φ_λ we have

$$\varphi_\lambda = \frac{q_2}{1 - q_2 - 2q_3},$$

and $\delta_\lambda = \varphi_\lambda/\lambda$.

Finally, we solve for φ_i, the frequency of the i-th alphanumeric character:

$$\begin{aligned}
\varphi_i &= \Phi_i(e) \\
&= q_1\Phi_i(I) + q_2[\Phi_i(I) + \Phi_i(E)] + 2q_3\Phi_i(E) \\
&= (q_1+q_2)\Phi_i(I) + (q_2+2q_3)\varphi_i.
\end{aligned}$$

Solving for φ_i we have

$$\varphi_i = \frac{(q_1 + q_2)\Phi_i(I)}{1 - q_2 - 2q_3}.$$

It remains to determine $\Phi_i(I)$:

$$\begin{aligned}
\Phi_i(I) &= \Phi_i\{(p_1\mathrm{a} \oplus \cdots \oplus p_{36}9)^{p+}\} \\
&= \Phi_i\{p_1\mathrm{a} \oplus \cdots \oplus p_{36}9\}/\bar{p} \\
&= [p_1\Phi_i(\mathrm{a}) + \cdots + p_{36}\Phi_i(9)]/\bar{p} \\
&= p_i/\bar{p}.
\end{aligned}$$

Substituting this into the equation for φ_i yields

$$\varphi_i = \frac{(q_1 + q_2)p_i}{(1 - q_2 - 2q_3)\bar{p}}.$$

As usual, $\delta_i = \varphi_i / \lambda$.

Unfortunately, the equations derived for the lambda-calculus are quite difficult to solve. In practice they would probably have to be solved numerically.

We can gain some insight into these equations by considering their behavior in some typical situations. Therefore, suppose that all the identifier characters are equally likely:

$$p_1 = p_2 = \cdots = p_{63} = 1/63.$$

Then we have

$$\varphi_i = \frac{q_1 + q_2}{63\bar{p}(1 - q_2 - 2q_3)}.$$

Now let $\mu = 1-q_2-2q_3$. Then we have

$$\lambda = [(q_1+q_2)/\bar{p} + 3q_2 + 2q_3]/\alpha,$$

$$\varphi_\lambda = q_2/\alpha,$$

$$\varphi_{lp} = \varphi_{rp} = (q_2+q_3)/\alpha,$$

$$\varphi_i = (q_1+q_2)/(63\bar{p}\alpha).$$

Rewriting the first equation:

$$\begin{aligned}\lambda &= [(q_1+q_2)/\bar{p}\alpha] + (3q_2 + 3q_3)/\alpha \\ &= \varphi_i/63 + (3q_2 + 2q_3)/\alpha \\ &= \varphi_i/63 + q_2/\alpha + (2q_2 + 2q_3)/\alpha \\ &= \varphi_i/63 + \varphi_\lambda + 2\varphi_{lp}.\end{aligned}$$

If we rewrite this

$$\lambda = \varphi_i/63 + \varphi_\lambda + \varphi_{lp} + \varphi_{rp},$$

then it becomes intuitive: the average length of a string is the sum of the [illegible] frequencies of occurrence of each terminal symbol.

Finally, we derive the information consumed per production by this grammar. To do this we expand the Mecne cross:

$$I \rightarrow \bar{p}AI \oplus pAU$$

$$A \rightarrow p_1 a \oplus \cdots p_{26} 9$$

and compute its average derivation length:

$$D(I) = 1 + \bar{p}D(A) + pD(A) + pD(I)$$

$$D(A) = 1$$

Therefore, $D(I) = 2 + pD(I)$, so $D(I) = 2/\bar{p}$. Next we compute $D(E)$:

$$D(E) = 1 + q_1 D(I) + q_2[D(I) + D(E)] + q_3[2D(E)]$$

$$= 1 + 2(q_1+q_2) + (q_2+2q_3)D(E).$$

Therefore,

$$D(E) = \frac{1 + 2(q_1+q_2)}{1 - q_2 - 2q_3} \text{ productions.}$$

The average information used per production is the ratio $H(E)/D(E)$, which is

$$Q = \frac{H(q_1,q_2,q_3) + (q_1+q_2)[H(p,\bar{p}) + H(p_1, \ldots, p_{26})]/\bar{p}}{1 + 2(q_1+q_2)} \text{ bits/production.}$$

7. Conclusions

We have described means for computing a number of information-theoretic properties of languages and their grammars. These properties include, for languages, their entropy, average string length, information density and density of occurrence for a given token. For grammars we have shown how to compute average derivation length and the information used by the grammar per production.

All of these techniques are based on the application of simple recursive formulas to

annotated grammars, grammars annotated with production probabilities. [illegible] that the same techniques can be applied to the computation of many other properties of both grammars and other symbol systems.

8. Acknowledgements

Work reported herein was supported in part by the Office of Naval Research under contract number N00014-84-WR-24087.

9. References

[1] Brillouin, Leon. *Science and Information Theory*, 2nd ed., New York: Academic Press, 1962.

[2] Brillouin, Leon. *Scientific Uncertainty, and Information*, New York: Academic Press, 1964.

[3] Cherry, Colin. *On Human Communication*, 2nd ed., Cambridge: The M.I.T. Press, 1975.

[4] Ginsburg, Seymour. *The Mathematical Theory of Context-Free Languages*, New York: McGraw-Hill, 1966.

[5] Hamming, R. W. *Coding and Information Theory*, Englewood Cliffs: Prentice-Hall, 1980.

[6] Hopcroft, J. E. and Ullman, J. D. *Formal Languages and their Relation to Automata*, Reading: Addison-Wesley, 1969.

[7] MacKay, Donald M. *Information, Mechanism and Meaning*, Cambridge: The M.I.T. Press, 1969.

[8] Shannon, C. E. A Mathematical Theory of Communication, *Bell Syst. Tech. J. 27*, 1948, pp 379-423; 623-656.

INITIAL DISTRIBUTION LIST

Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
Dudley Knox Library Code 0142 Naval Postgraduate School Monterey, CA 93943	2
Office of Research Administration Code 012A Naval Postgraduate School Monterey, CA 93943	1
Associate Professor B. J. MacLennan, Code 52Ml Department of Computer Science Naval Postgraduate School Monterey, CA 93943	12
Chairman, Code 52Hq Department of Computer Science Naval Postgraduate School Monterey, CA 93943	40
Dr. Robert Grafton Code 433 Office of Naval Research 800 N. Quincy Arlington, VA 22217	1
Dr. David W. Mizell Office of Naval Research 1030 East Green Street Pasadena, CA 91106	1
Professor Robert Floyd Department of Computer Science Stanford University Stanford, CA	1

Professor Pranas Zunde 1
School of Information - and Computer Science
Georgia Institute of Technology
Atlanta, GA 30332

Professor Jaakko Hintikka 1
Department of Philosophy
Florida State University
Tallahassee, FL 32306

DUDLEY KNOX LIBRARY
3 2768 00332759 4